UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/822,938	03/30/2001	Ronny Ronen	2207/8610	8312

26646 7590 01/29/2004

KENYON & KENYON
ONE BROADWAY
NEW YORK, NY 10004

EXAMINER	
GERSTL, SHANE F	

ART UNIT	PAPER NUMBER
2183	6

DATE MAILED: 01/29/2004

Please find below and/or attached an Office communication concerning this application or proceeding.

Office Action Summary	Application No.	Applicant(s)	
	09/822,938	RONEN ET AL.	
	Examiner	Art Unit	
	Shane F Gerstl	2183	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If the period for reply specified above is less than thirty (30) days, a reply within the statutory minimum of thirty (30) days will be considered timely.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133).
- Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *13 November 2002 and 17 July 2001*.

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-28* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-28* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☒ The specification is objected to by the Examiner.

10)☒ The drawing(s) filed on *30 March 2001* is/are: a)☐ accepted or b)☒ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☒ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. §§ 119 and 120

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

13)☐ Acknowledgment is made of a claim for domestic priority under 35 U.S.C. § 119(e) (to a provisional application) since a specific reference was included in the first sentence of the specification or in an Application Data Sheet. 37 CFR 1.78.

a) ☐ The translation of the foreign language provisional application has been received.

14)☐ Acknowledgment is made of a claim for domestic priority under 35 U.S.C. §§ 120 and/or 121 since a specific reference was included in the first sentence of the specification or in an Application Data Sheet. 37 CFR 1.78.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☒ Information Disclosure Statement(s) (PTO-1449) Paper No(s) 3.
4) ☐ Interview Summary (PTO-413) Paper No(s). _____.
5) ☐ Notice of Informal Patent Application (PTO-152)
6) ☐ Other:

DETAILED ACTION

1. Claim1-28 have been examined.

Papers Received

2. Receipt is acknowledged of information disclosure statement, declaration, and request for corrected filing receipt papers submitted, where the papers have been placed of record in the file.

Oath/Declaration

3. Applicant has not given a post office address anywhere in the application papers as required by 37 CFR 1.33(a), which was in effect at the time of filing of the oath or declaration. A statement over applicant's signature providing a complete post office address is required.

Drawings

4. The drawings are objected to as failing to comply with 37 CFR 1.84(p)(5) because they do not include the following reference sign(s) mentioned in the description: 289 and 300. A proposed drawing correction or corrected drawings are required in reply to the Office action to avoid abandonment of the application. The objection to the drawings will not be held in abeyance.

5. The drawings are objected to as failing to comply with 37 CFR 1.84(p)(5) because they include the following reference sign(s) not mentioned in the description: 285, 410, 420, and 430. A proposed drawing correction, corrected drawings, or amendment to the specification to add the reference sign(s) in the description, are

required in reply to the Office action to avoid abandonment of the application. The objection to the drawings will not be held in abeyance.

6. The drawings are objected to as failing to comply with 37 CFR 1.84(p)(4) because reference characters "280" (in the specification) and "285" (in the figure) have both been used to designate the wide multiplexer of figure 2. A proposed drawing correction or corrected drawings are required in reply to the Office action to avoid abandonment of the application. The objection to the drawings will not be held in abeyance.

7. The drawings are objected to because the figure titles and reference characters used have been informally drawn in ink by hand, which leads to some difficulty in reading them. The shading of the drawings also makes the drawings hard to read. A proposed drawing correction or corrected drawings are required in reply to the Office action to avoid abandonment of the application. The objection to the drawings will not be held in abeyance.

Specification

8. The title of the invention is not descriptive. A new title is required that is clearly indicative of the invention to which the claims are directed.

The following title is suggested: Apparatus, Method, and System for Fast Register Renaming Using Virtual Renaming by way of Choosing Either Rename Information or a Renamed Register.

Claim Rejections - 35 USC § 102

9. The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

> A person shall be entitled to a patent unless –
>
> (b) the invention was patented or described in a printed publication in this or a foreign country or in public use or on sale in this country, more than one year prior to the date of application for patent in the United States.

10. Claims 1-28 rejected under 35 U.S.C. 102(b) as being anticipated by Vajapeyam et al.

11. In regard to claim 1, Vajapeyam has disclosed a method for renaming a source for use with a processor, the method comprising:

a. providing at least one instruction; Page 1, column 2 shows that instructions are fetched and decoded and thus provided.

b. building instruction dependency information based on the at least one instruction; In the same paragraph as cited directly above, it is shown that these fetched and decoded instructions are renamed. Since renaming is used to resolve dependencies (as shown in the abstract as the underlying goal) the dependency information is gathered or built to accommodate this renaming.

c. caching the at least one instruction with the instruction dependency information to provide cached instruction information; Page 2, column 2 shows that register renaming (dependency) information is recorded in the trace cache. Figure 5 shows that the trace cache entries hold the instruction and the associated dependency information.

d. renaming a register based on the cached instruction information to provide a renamed register; Page 5, section 2.3 shows that a lower-bandwidth single

cycle rename stage processes information from the cache. One of ordinary skill in the art would recognize that the rename stage inherently produces a renamed register.

e. And multiplexing the instruction dependency information and the renamed register to rename the source. Page 4, column 2 shows that a local mapping is forwarded a subsequent map modification stage or a mapping is picked up from a global free list (table). It is shown here that the selection between these two options is made by a multiplexer. The mapping that is picked up from the global free list is the renamed register as described by the process for traditional register renaming at the top of the paragraph. The local mapping is dependency information for the instruction that is sent to another stage for modification before global renaming.

12. In regard to claim 2, Vajapeyam discloses the method of claim 1, wherein the cached instruction information includes first source information, second source information and destination information (figure 5).

13. In regard to claim 3, Vajapeyam discloses the method of claim 2, wherein the first source information includes three bits, the second source information includes three bits, and the destination information includes one bit. Figure 5 shows that the first and second source information contain 4 fields and thus, logically, include at least three bits. The figure also shows that the destinations information includes 3 fields and therefore includes at least one bit.

14. In regard to claim 4, Vajapeyam discloses the method of claim 1, wherein the first source information and the second source information denote a rename window instruction from which the at least one instruction depends. Applicant has not defined what a rename window actually is in the specification, but merely uses it. Therefore, the examiner is taking a rename window instruction to be any instruction containing two sources with another instruction depending on it as the claim suggests. Vajapeyam shows that all instructions have two sources as shown in figure 5. Since these instructions were renamed, another instruction must have a dependency on each one and is thus a rename window instruction (or trace window instructions as shown in figure 2).

15. In regard to claim 5, Vajapeyam discloses the method of claim 1, wherein a virtual ID is formed by at least one of adding, combining, concatenating or logically OR-ing upper bits from a renamer with lower bits from an instruction cache. Figure 6 shows the renamer interacting with a line from the trace cache (instruction cache). The trace cache sends data to a MAP TBL (part of the renamer) where based on the data it receives, a renamed register or virtual ID is formed. Thus, essentially, bits (lower bits) from the cache are combined with bits (upper bits) from the renamer.

16. In regard to claim 6, Vajapeyam discloses the method of claim 1, wherein:

a. the first source information includes three bits, the second source information includes three bits, and the destination information includes one bit; Figure 5 shows that the first and second source information contain 4 fields and

thus, logically, include at least three bits. The figure also shows that the destinations information includes 3 fields and therefore includes at least one bit.

b. the first source information and the second source information denote a rename window instruction from which the at least one instruction depends; Applicant has not defined what a rename window actually is in the specification, but merely uses it. Therefore, the examiner is taking a rename window instruction to be any instruction containing two sources with another instruction depending on it. All instructions have two sources as shown in figure 5 of Vajapeyam. Since these instructions were renamed, another instruction must have a dependency on each one and is thus a rename window instruction (or trace window instructions as shown in figure 2).

c. and a virtual ID is formed by at least one of adding, combining, concatenating or logically OR-ing upper bits from a renamer with lower bits from an instruction cache. Figure 6 shows the renamer interacting with a line from the trace cache (instruction cache). The trace cache sends data to a MAP TBL (part of the renamer) where based on the data it receives, a renamed register or virtual ID is formed. Thus, essentially, bits (lower bits) from the cache are combined with bits (upper bits) from the renamer.

17. In regard to claim 7, Vajapeyam discloses the method of claim 6, wherein the processor is a microprocessor. The IEEE dictionary definition included herein defines a "microprocessor" to be "an integrated circuit that contains the logic elements for

manipulating data and for making decisions." Thus the superscalar processor taught by Vajapeyam is a microprocessor.

18. In regard to claim 8, Vajapeyam discloses a system for renaming a source for use with a processor, the system comprising:

a. a fetch and decoding arrangement for fetching and decoding at least one instruction from the processor (figure 2);

b. a build-instruction-dependency arrangement for building instruction dependency information based on the at least one instruction; On page 1, column 2, it is shown that these fetched and decoded instructions are renamed. Since renaming is used to resolve dependencies (as shown in the abstract as the underlying goal) the dependency information is gathered or built to accommodate this renaming.

c. an instruction cache arrangement for caching the at least one instruction with the instruction dependency information to provide cached instruction information, the build-instruction-dependency arrangement providing the instruction dependency information to the instruction cache arrangement; Page 2, column 2 shows that register renaming (dependency) information is recorded in a trace cache (of figure 2). Figure 5 shows that the trace cache entries hold the instruction and the associated dependency information.

d. a renamer arrangement for renaming a register based on the cached instruction information and for providing a renamed register; Figure 2 shows a

remap unit (renamer) that renames registers of instructions fetched from the cache.

e. and a multiplexing arrangement for multiplexing the instruction dependency information and the renamed register and for providing a renamed source. Page 4, column 2 shows that a local mapping is forwarded a subsequent map modification stage or a mapping is picked up from a global free list (table). It is shown here that the selection between these two options is made by a multiplexer. The mapping that is picked up from the global free list is the renamed register as described by the process for traditional register renaming at the top of the paragraph. The local mapping is dependency information for the instruction that is sent to another stage for modification before global renaming.

19. In regard to claim 9, Vajapeyam discloses the system of claim 8, wherein the cached instruction information includes first source information, second source information and destination information (figure 5).

20. In regard to claim 10, Vajapeyam discloses the system of claim 9, wherein the first source information includes three bits, the second source information includes three bits, and the destination information includes one bit. Figure 5 shows that the first and second source information contain 4 fields and thus, logically, include at least three bits. The figure also shows that the destinations information includes 3 fields and therefore includes at least one bit.

21. In regard to claim 11, Vajapeyam discloses the system of claim 8, wherein the first source information and the second source information denote a rename window

instruction from which the at least one instruction depends. Applicant has not defined what a rename window actually is in the specification, but merely uses it. Therefore, the examiner is taking a rename window instruction to be any instruction containing two sources with another instruction depending on it. All instructions have two sources as shown in figure 5 of Vajapeyam. Since these instructions were renamed, another instruction must have a dependency on each one and is thus a rename window instruction (or trace window instructions as shown in figure 2).

22. In regard to claim 12, Vajapeyam discloses the system of claim 8, wherein a virtual ID is formed by at least one of adding, combining, concatenating or logically OR-ing upper bits from a renamer with lower bits from an instruction cache. Figure 6 shows the renamer interacting with a line from the trace cache (instruction cache). The trace cache sends data to a MAP TBL (part of the renamer) where based on the data it receives, a renamed register or virtual ID is formed. Thus, essentially, bits (lower bits) from the cache are combined with bits (upper bits) from the renamer.

23. In regard to claim 13, Vajapeyam discloses the system of claim 8, wherein:

a. the first source information includes three bits, the second source information includes three bits, and the destination information includes one bit; Figure 5 shows that the first and second source information contain 4 fields and thus, logically, include at least three bits. The figure also shows that the destinations information includes 3 fields and therefore includes at least one bit.

b. the first source information and the second source information denote a rename window instruction from which the at least one instruction depends;

Applicant has not defined what a rename window actually is in the specification, but merely uses it. Therefore, the examiner is taking a rename window instruction to be any instruction containing two sources with another instruction depending on it. All instructions have two sources as shown in figure 5 of Vajapeyam. Since these instructions were renamed, another instruction must have a dependency on each one and is thus a rename window instruction (or trace window instructions as shown in figure 2).

c. and a virtual ID is formed by at least one of adding, combining, concatenating or logically OR-ing upper bits from a renamer with lower bits from an instruction cache. Figure 6 shows the renamer interacting with a line from the trace cache (instruction cache). The trace cache sends data to a MAP TBL (part of the renamer) where based on the data it receives, a renamed register or virtual ID is formed. Thus, essentially, bits (lower bits) from the cache are combined with bits (upper bits) from the renamer.

24. In regard to claim 14, Vajapeyam discloses the method of claim 13, wherein the processor is a microprocessor. The IEEE dictionary definition included herein defines a "microprocessor" to be "an integrated circuit that contains the logic elements for manipulating data and for making decisions." Thus the superscalar processor taught by Vajapeyam is a microprocessor.

25. In regard to claim 15, Vajapeyam discloses a system for renaming a source for use with a processor, the system comprising:

a. Means for fetching and decoding at least one instruction from the processor (figure 2);

b. Means for building instruction dependency information based on the at least one instruction; On page 1, column 2, it is shown that these fetched and decoded instructions are renamed. Since renaming is used to resolve dependencies (as shown in the abstract as the underlying goal) the dependency information is gathered or built to accommodate this renaming.

c. Means for caching the at least one instruction with the instruction dependency information to provide cached instruction information, the build-instruction-dependency arrangement providing the instruction dependency information to the instruction cache arrangement; Page 2, column 2 shows that register renaming (dependency) information is recorded in a trace cache (of figure 2). Figure 5 shows that the trace cache entries hold the instruction and the associated dependency information.

d. Means for renaming a register based on the cached instruction information and for providing a renamed register; Figure 2 shows a remap unit (renamer) that renames registers of instructions fetched from the cache.

e. And means for multiplexing the instruction dependency information and the renamed register and for providing a renamed source. Page 4, column 2 shows that a local mapping is forwarded a subsequent map modification stage or a mapping is picked up from a global free list (table). It is shown here that the selection between these two options is made by a multiplexer. The mapping that

is picked up from the global free list is the renamed register as described by the process for traditional register renaming at the top of the paragraph. The local mapping is dependency information for the instruction that is sent to another stage for modification before global renaming.

26. In regard to claim 16, Vajapeyam discloses the system of claim 15, wherein the cached instruction information includes first source information, second source information and destination information (figure 5).

27. In regard to claim 17, Vajapeyam discloses the system of claim 16, wherein the first source information includes three bits, the second source information includes three bits, and the destination information includes one bit. Figure 5 shows that the first and second source information contain 4 fields and thus, logically, include at least three bits. The figure also shows that the destinations information includes 3 fields and therefore includes at least one bit.

28. In regard to claim 18, Vajapeyam discloses the system of claim 15, wherein the first source information and the second source information denote a rename window instruction from which the at least one instruction depends. Applicant has not defined what a rename window actually is in the specification, but merely uses it. Therefore, the examiner is taking a rename window instruction to be any instruction containing two sources with another instruction depending on it. All instructions have two sources as shown in figure 5 of Vajapeyam. Since these instructions were renamed, another instruction must have a dependency on each one and is thus a rename window instruction (or trace window instructions as shown in figure 2).

29. In regard to claim 19, Vajapeyam discloses the system of claim 15, wherein a virtual ID is formed by at least one of adding, combining, concatenating or logically OR-ing upper bits from a renamer with lower bits from an instruction cache. Figure 6 shows the renamer interacting with a line from the trace cache (instruction cache). The trace cache sends data to a MAP TBL (part of the renamer) where based on the data it receives, a renamed register or virtual ID is formed. Thus, essentially, bits (lower bits) from the cache are combined with bits (upper bits) from the renamer.

30. In regard to claim 20, Vajapeyam discloses the system of claim 15, wherein:

a. the first source information includes three bits, the second source information includes three bits, and the destination information includes one bit; Figure 5 shows that the first and second source information contain 4 fields and thus, logically, include at least three bits. The figure also shows that the destinations information includes 3 fields and therefore includes at least one bit.

b. the first source information and the second source information denote a rename window instruction from which the at least one instruction depends; Applicant has not defined what a rename window actually is in the specification, but merely uses it. Therefore, the examiner is taking a rename window instruction to be any instruction containing two sources with another instruction depending on it. All instructions have two sources as shown in figure 5 of Vajapeyam. Since these instructions were renamed, another instruction must have a dependency on each one and is thus a rename window instruction (or trace window instructions as shown in figure 2).

c. and a virtual ID is formed by at least one of adding, combining, concatenating or logically OR-ing upper bits from a renamer with lower bits from an instruction cache. Figure 6 shows the renamer interacting with a line from the trace cache (instruction cache). The trace cache sends data to a MAP TBL (part of the renamer) where based on the data it receives, a renamed register or virtual ID is formed. Thus, essentially, bits (lower bits) from the cache are combined with bits (upper bits) from the renamer.

31. In regard to claim 21, Vajapeyam discloses the method of claim 15, wherein the processor is a microprocessor. The IEEE dictionary definition included herein defines a "microprocessor" to be "an integrated circuit that contains the logic elements for manipulating data and for making decisions." Thus the superscalar processor taught by Vajapeyam is a microprocessor.

32. In regard to claim 22, the examiner is taking "a storage medium" to be a computer-readable medium because the set of instructions does not appear to have any other utility in this instance than in a computer and is thus statutory. Vajapeyam has disclosed a set of instructions residing in a storage medium, said set of instructions capable of being executed by a processor to implement a method for renaming a source for use with a processor, (Vajapeyam's disclosed processor uses instructions stored in a cache as shown below) the method comprising:

a. providing at least one instruction; Page 1, column 2 shows that instructions are fetched and decoded and thus provided.

b. building instruction dependency information based on the at least one instruction; In the same paragraph as cited directly above, it is shown that these fetched and decoded instructions are renamed. Since renaming is used to resolve dependencies (as shown in the abstract as the underlying goal) the dependency information is gathered or built to accommodate this renaming.

c. caching the at least one instruction with the instruction dependency information to provide cached instruction information; Page 2, column 2 shows that register renaming (dependency) information is recorded in the trace cache. Figure 5 shows that the trace cache entries hold the instruction and the associated dependency information.

d. renaming a register based on the cached instruction information to provide a renamed register; Page 5, section 2.3 shows that a lower-bandwidth single cycle rename stage processes information from the cache. One of ordinary skill in the art would recognize that the rename stage inherently produces a renamed register.

e. And multiplexing the instruction dependency information and the renamed register to rename the source. Page 4, column 2 shows that a local mapping is forwarded a subsequent map modification stage or a mapping is picked up from a global free list (table). It is shown here that the selection between these two options is made by a multiplexer. The mapping that is picked up from the global free list is the renamed register as described by the process for traditional register renaming at the top of the paragraph. The local mapping is dependency

information for the instruction that is sent to another stage for modification before global renaming.

33. In regard to claim 23, Vajapeyam discloses the method of claim 22, wherein the cached instruction information includes first source information, second source information and destination information (figure 5).

34. In regard to claim 24, Vajapeyam discloses the method of claim 23, wherein the first source information includes three bits, the second source information includes three bits, and the destination information includes one bit. Figure 5 shows that the first and second source information contain 4 fields and thus, logically, include at least three bits. The figure also shows that the destinations information includes 3 fields and therefore includes at least one bit.

35. In regard to claim 25, Vajapeyam discloses the method of claim 22, wherein the first source information and the second source information denote a rename window instruction from which the at least one instruction depends. Applicant has not defined what a rename window actually is in the specification, but merely uses it. Therefore, the examiner is taking a rename window instruction to be any instruction containing two sources with another instruction depending on it. All instructions have two sources as shown in figure 5 of Vajapeyam. Since these instructions were renamed, another instruction must have a dependency on each one and is thus a rename window instruction (or trace window instructions as shown in figure 2).

36. In regard to claim 26, Vajapeyam discloses the method of claim 22, wherein a virtual ID is formed by at least one of adding, combining, concatenating or logically OR-

ing upper bits from a renamer with lower bits from an instruction cache. Figure 6 shows the renamer interacting with a line from the trace cache (instruction cache). The trace cache sends data to a MAP TBL (part of the renamer) where based on the data it receives, a renamed register or virtual ID is formed. Thus, essentially, bits (lower bits) from the cache are combined with bits (upper bits) from the renamer.

37. In regard to claim 27, Vajapeyam discloses the method of claim 22, wherein:

a. the first source information includes three bits, the second source information includes three bits, and the destination information includes one bit; Figure 5 shows that the first and second source information contain 4 fields and thus, logically, include at least three bits. The figure also shows that the destinations information includes 3 fields and therefore includes at least one bit.

b. the first source information and the second source information denote a rename window instruction from which the at least one instruction depends; Applicant has not defined what a rename window actually is in the specification, but merely uses it. Therefore, the examiner is taking a rename window instruction to be any instruction containing two sources with another instruction depending on it. All instructions have two sources as shown in figure 5 of Vajapeyam. Since these instructions were renamed, another instruction must have a dependency on each one and is thus a rename window instruction (or trace window instructions as shown in figure 2).

c. and a virtual ID is formed by at least one of adding, combining, concatenating or logically OR-ing upper bits from a renamer with lower bits from

an instruction cache. Figure 6 shows the renamer interacting with a line from the trace cache (instruction cache). The trace cache sends data to a MAP TBL (part of the renamer) where based on the data it receives, a renamed register or virtual ID is formed. Thus, essentially, bits (lower bits) from the cache are combined with bits (upper bits) from the renamer.

38. In regard to claim 28, Vajapeyam discloses the method of claim 27, wherein the processor is a microprocessor. The IEEE dictionary definition included herein defines a "microprocessor" to be "an integrated circuit that contains the logic elements for manipulating data and for making decisions." Thus the superscalar processor taught by Vajapeyam is a microprocessor.

Conclusion

39. The following is text cited from 37 CFR 1.111(c): In amending in reply to a rejection of claims in an application or patent under reexamination, the applicant or patent owner must clearly point out the patentable novelty which he or she thinks the claims present in view of the state of the art disclosed by the references cited or the objections made. The applicant or patent owner must also show how the amendments avoid such references or objections.

40. The prior art made of record and not relied upon is considered pertinent to applicant's disclosure. The following publication and patents have been cited to further show the art with respect to caches and register renaming in general.

Aggressive dynamic execution of multimedia kernel to Bishop et al. discusses a trace cache of decoded and preprocessed instructions that can be quickly renamed or a slower process for a trace cache miss.

US Pat No 5,812,812 to Afsar has an instruction cache that includes a cache that holds dependency information for renaming.

US Pat No 5,337,415 to DeLano discloses storing instructions with predecoded dependency information in the cache for renaming.

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Shane F Gerstl whose telephone number is (703)305-7305. The examiner can normally be reached on M-F 6:45-4:15 (First Friday Off).

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Eddie Chan can be reached on (703)305-9712. The fax phone number for the organization where this application or proceeding is assigned is (703)872-9306.

Any inquiry of a general nature or relating to the status of this application or proceeding should be directed to the receptionist whose telephone number is (703)305-3900.

Shane F Gerstl
Examiner
Art Unit 2183

SFG
January 15, 2004

EDDIE CHAN
SUPERVISORY PATENT EXAMINER
TECHNOLOGY CENTER 2100